Multiagentensimulation sozialer Phänomene: eine praktische Einführung

Pfeffer, Jürgen

Veröffentlichungsversion / Published Version
Zeitschriftenartikel / journal article

Multiagentensimulation sozialer Phänomene: Eine praktische Einführung [1]

Jürgen Pfeffer

Zusammenfassung

Simulationen sind Nachbildungen von Abläufen in realen und meist komplexen Systemen. Multiagentensimulationen beschreiben das Verhalten der einzelnen Akteure dieser Systeme. In den Sozialwissenschaften finden Multiagentensimulationen zunehmend Verbreitung (Gilbert & Troitzsch 2005, Gilbert 2007, Miller et al. 2007). Dies ist vor allem durch die fortschreitenden Entwicklungen im Bereich der Homecomputer ermöglicht, da Multiagentensimulationen ohne moderne EDV-Infrastruktur undenkbar wäre. Dieser Artikel gibt im ersten Teil eine Einführung in die Theorie der Modellbildung und Simulation sowie einen Abschnitt zur Simulation als virtuelles Experiment. Der zweite Teil dient der praktischen Einführung in die Simulationsumgebung StarLogo. Simulationsumgebungen werden dabei als inhaltsneutrale, computerbasierte Tools verstanden, in denen komplexe, reale Systeme nachgebildet und simuliert werden können. Ziel des vorliegenden Artikels ist es, Sozialwissenschafterinnen und Sozialwissenschafter ohne Erfahrungen aus dem Bereich der Programmierung in die Programmerstellung von Multiagentensimulationen einzuführen und zur Umsetzung von eigenen Projekten anzuregen und zu motivieren.

Schlagworte: Multiagentensysteme, Modellbildung, Simulation, StarLogo

Modellbildung

Ein *Modell* wird im Kontext der Computersimulation nach der Richtlinie 3633 des Vereins Deutscher Ingenieure e.V. als „vereinfachte Nachbildung eines geplanten oder real existierenden Systems“ (VDI 1996) verstanden. *Simulation* wird in der gleichen Richtlinie als „das Nachbilden eines Systems mit seinen dynamischen Prozessen in einem experimentierfähigen Modell, um zu Erkenntnissen zu gelangen, die auf die Wirklichkeit übertragbar sind“ definiert. Im Kontext der Simulation sozialer Phänomene bildet ein Modell demnach ein reales System nach, das aus Akteuren und deren Verhalten besteht. Durch den Vorgang der Nachbildung eines mentalen Modells in eine mathematische Struktur und ihrer Simulation in Simulationsumgebungen entstehen neue Erkenntnisse über das zugrundeliegende reale System. Von besonderer Bedeutung sind dynamische Modelle, in denen unterschiedliche zeitliche Veränderungen beobachtet werden können. So können Prozesse der Wirtschaftsentwicklung in wenigen Minuten betrachtet werden, die in der Wirklichkeit Jahrzehnte brauchen würden, aber auch molekulare Bewegungen, die sich in der Realität in Mikrosekunden abspielen, menschlichen Beobachtungsmöglichkeiten entsprechend in der Zeit gedehnt und verlangsamt werden. In manchen Fällen können die provisorischen Resultate der Simulation mit der Realität verglichen werden. Sie erlauben eine Art Nachbildung eines Experiments, in der die Umwelt gezielt manipuliert wird und die daraus entstehenden Veränderungen mehr oder weniger direkt beobachtet werden können. Die Simulationen können für unterschiedlichste fachliche Fragestellungen angewendet werden: von politischen, ökonomischen, ökologischen, sozialen, psychologischen, rechtlichen, künstlerischen, religösen Bereichen bis zu physikalischen, chemischen, biologischen und physiologischen Untersuchungen.

---

1 Teile dieses Artikels werden auch publiziert in Pfeffer, Jürgen; Fleissner, Peter (2009) Modellbildung, in: Christian Stegbauer, Roger Häußling (Hrsg), Handbuchs Netzwerkforschung, Wiesbaden: VS-Verlag.

Eine Beschreibung eines Modells wird 1973 von Herbert Stachowiak in seiner *allgemeinen Modelltheorie* (Stachowiak 1973) vorgenommen. Demnach zeichnet ein Modell drei Kriterien aus:

1. Abbildung. Ein Modell ist immer eine Repräsentation eines realen Systems.
2. Verkürzung. Es ist weder möglich noch erwünscht, alle Eigenschaften des Originals abzubilden. Es wird versucht, die für die Fragestellung oder Aufgabenstellung relevanten Eigenschaften zu extrahieren.
3. Pragmatismus. Ein Modell steht nicht für sich selbst, sondern muss interpretiert werden. Diese Interpretation orientiert sich an der Nützlichkeit für die ModelliererInnen.

Für die Generierung von Modellen bedeutet das, dass ein Modell zwar einerseits die Struktur der Wirklichkeit nachbilden soll, aber andererseits auch klarstellt, dass diese Nachbildung nicht umfassend sein kann und auch nicht sein soll. Modelle sind also Nachbildungen und gleichzeitig Entwürfe, die „… eine wirklichkeitsnahe, jedoch einfachere, billigere oder ungefährlichere Untersuchung als das Objekt erlauben." (Brockhaus 1983).

Multiagentensimulation

Im Zentrum von Multiagentensystemen stehen die einzelnen Akteure eines Systems und ihr Verhalten. Die Beschreibung der Akteure erfolgt auf lokaler (Mikro-)Ebene. Durch das lokale Wirken der Akteure kann auf der Makroebene eine qualitative Veränderung des Systems entstehen (*Emergenz*), die aus dem Verhalten der einzelnen Akteure nicht direkt vorhersehbar bzw. ableitbar ist. Multiagentensysteme werden für die Nachbildung von Systemen verwendet, die sich aus vielen Akteuren zusammensetzen (z.B. Verhalten von Ameisenkolonien, die Verbreitung von Informationen oder Krankheiten in der Bevölkerung).

Der Grundgedanke, auf dem Simulationen mit Multiagentensystemen beruhen, ist jener der Dezentralisierung und Individualisierung (Resnick 1994). Steven Johnson (Johnson 2001) beschreibt verschiedene Phänomene der Emergenz. Darunter versteht man das Entstehen von Strukturen aufgrund lokalen Verhaltens. Ausgangspunkt seiner Schilderungen sind dabei die Forschungen von Deborah Gordon über das Verhalten von Ameisen (Gordon 1999). Gordon untersuchte über mehrere Jahre rote Ernteameisen (Pogonomyrmex barbatus) in der Wüste von Arizona und beobachtete dabei neben dem Verhalten der einzelnen Ameisen vor allem Eigenschaften und Verhalten unterschiedlicher Ameisenkolonien.

Die Simulation einzelner Akteure auf der Mikroebene mit relativ einfachem lokalem Verhalten und die Beobachtung der dadurch entstehenden Veränderung des Gesamtsystems auf der Makroebene sind die Kernelemente der Multiagentensimulation. Als *Agent* in solchen Systemen bezeichnet man die Repräsentation einer realen Einheit (Ameise, Mensch, Firma,…) innerhalb eines Computerprogramms (Gilbert 2007). Eine wichtige Eigenschaft dieser Agenten ist die Interaktion mit anderen Agenten oder der Umwelt. Das Verhalten dieser Akteure folgt in jedem Simulationsschritt eindeutigen und meist nach einfachen Regeln.

Modellbildung und Simulation als virtuelles Experiment

Eine zentrale Aufgabe der Modellbildung ist es, Einsicht in das zugrundeliegende reale System zu erlangen. Mit Modellen in einer Simulationsumgebung experimentieren, indem verschiedene Pa-

rameter verändert und die dabei entstandene Auswirkung auf einzelne Elemente oder das gesamte System beobachtet werden, hat einen spielerischen Charakter (vgl. z.B. Colella et al. 2001). Gleichzeitig ist dies mit dem Sammeln von Erfahrungen über das System verbunden und führt so zu Erkenntnisgewinn. In Verbindung mit einer Fragestellung, die mit Hilfe des Modells beantwortet werden soll, wird aus dem spielerischen Experimentieren ein Experiment, das dem aus der naturwissenschaftlichen Forschung bekannten und etablierten Experiment in vielem gleicht. Im Folgenden sollen die Übereinstimmungen sowie die Unterscheidungen zwischen herkömmlichen naturwissenschaftlichen Experimenten und Experimenten mit computerunterstützten Modellen herausgearbeitet werden.

Unter einem Experiment versteht man ein Untersuchungsdesign, in dem „der Forscher einzelne Bedingungsfaktoren (unabhängige Variablen) variiert, um zu sehen, welche Effekte (abhängige Variablen) sich daraus ergeben." (Kühl 2005). Als unabhängige Variablen werden dabei jene bezeichnet, die von den ForscherInnen „absichtsvoll und geplant" verändert werden. Durch diese gewollte Veränderung geschieht auch die Veränderung von anderen, den abhängigen Variablen. Als Hypothese bezeichnet man die Vorhersage dieses Effekts. Störvariablen sind jene, die diesen experimentell beobachteten Effekt verfälschen.

Von den verschiedenen Formen des Experiments entsprechen Experimente mit computerunterstützten Modellen der Definition von Laborexperimenten, in denen ein kontrolliertes Verändern der unabhängigen Variablen sowie eine Kontrolle der Störvariablen möglich sind.

Der Ablauf eines Experimentes folgt in unterschiedlichen wissenschaftlichen Bereichen im Kern dem folgenden Ablauf (Kirkup 1994):

- Festlegen eines Ziels, einer Hypothese
- Erstellen eines Umsetzungsplanes des Experiments
- Vorbereitungen und Organisation des Experiments
- Testphase des Experiments
- Durchführungsphase inklusive Datensammlung
- Wiederholung des Experiments
- Analyse der Daten
- Schlussfolgerungen inklusive Überprüfung der Hypothesen
- Erstellung eines Berichts

Diese Punkte finden sich auch in virtuellen Experimenten wieder, allerdings mit veränderter Bedeutung. Die Vorbereitungen sind weit weniger aufwändig und bestehen bei einem fertigen Modell nur aus der Einrichtung der Startwerte der Variablen. Die Testphase ist bei Simulationen im Computer nicht notwendig, da aufgrund der Geschwindigkeit, der Verfügbarkeit und der Kosten von Computern der eigentliche Simulationslauf ein unwesentlicher Aufwand ist. Die Datensammlung verläuft bei Computersimulationen in der Regel automatisch. Wiederholungen sind in Computersimulationen kein zusätzlicher Aufwand, daher werden in vielen wissenschaftlichen Projekten tausende Simulationsdurchläufe durchgeführt.

Einen relevanten Aspekt bei der Verwendung einer Computersimulation als virtuelles Experiment stellen die Schlussfolgerungen dar. Die folgende Abbildung ist eine Erweiterung der Logik der Simulation aus (Gilbert und Troitzsch 2005). Durch Abstraktion entsteht ein Modell. Durch Simulation werden Daten gewonnen, die mit den durch Beobachtung gesammelten realen Daten des Sys-

tems verglichen werden. Erweitert man die Simulation um eine Hypothese zu einem Experiment, wirkt diese ebenfalls auf die Simulation und die daraus erzielten Effekte. Diese Effekte, die in herkömmlichen Experimenten Rückschlüsse auf die Hypothese zulassen und zu einer Adaption dieser führen können, stehen aber auch in direkter Abhängigkeit zum zugrundeliegenden Modell, und sind durch die Grundannahmen der Abstraktion determiniert.

Einfach gesagt kann aus einem unerwarteten Ergebnis nicht klar gefolgert werden, ob die Hypothese oder das Modell falsch sind. Das Modell kann als eine Art Störvariable in Experimenten mit Simulationsumgebungen gesehen werden bzw. mit verzerrenden Einflussfaktoren, wie sie aus der sozialwissenschaftlichen Forschung bekannt sind (z.B. Interviewereffekt), verglichen werden. Tatsächlich findet die Abstraktion des realen Systems in ein Modell noch zusätzlich über den Umweg des mentalen Modells statt.

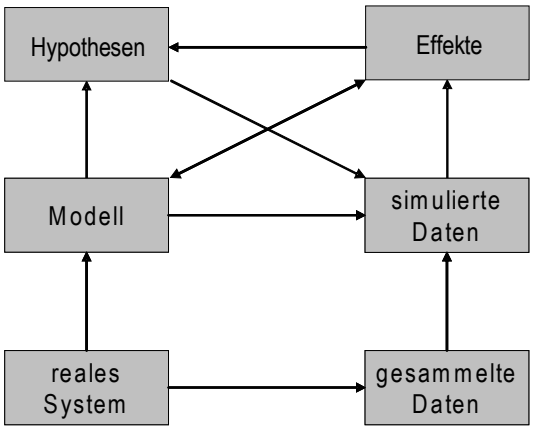

Abbildung 1: Logik des computersimulierten Experiments

Simulationsumgebungen als Hilfsmittel

Simulationsumgebungen sind, wie eingangs definiert, inhaltsneutrale, computerbasierte Tools, in denen komplexe, reale Systeme konstruiert und simuliert werden können. Simulationsumgebungen bieten die Möglichkeit, durch Nachbildung eines Systems in ein Modell und durch die darauffolgende Simulation dieses Modells, Erkenntnisse über das zugrundeliegende reale System zu gewinnen. Mit Hilfe von Simulationsumgebungen ist es möglich, unterschiedlichste komplexe Systeme zu erforschen. Bei der Konstruktion von Modellen in Simulationsumgebungen ist eine detaillierte Auseinandersetzung mit dem realen System und eine Fokussierung auf dessen essentielle Eigenschaften erforderlich.

Durch den Einsatz von Simulationsumgebungen, die auch ohne Programmierkenntnisse verwendet werden können (vgl. die Übersicht weiter hinten in diesem Artikel), öffnet sich die Multiagentensimulation vielen Forscherinnen und Forschern aus unterschiedlichsten Bereichen. Dabei ist ein unkomplizierter Einsatz ohne aufwändige Transaktionskosten von zentraler Bedeutung. Es soll nicht das technische Verständnis der Simulationsumgebung im Vordergrund stehen, sondern die

praktische Nutzung und der daraus entstehende Erkenntnisgewinn. Im Rahmen des vorliegenden Artikels ist die Simulationsumgebung StarLogo als Arbeitsinstrument gewählt worden. StarLogo ermöglicht einerseits, mit wenigen Befehlen innerhalb kurzer Zeit eigene Simulationen zu erstellen und ist weit verbreitet, wodurch eine große Anzahl an Einführungen und Benutzerhandbücher für unterschiedliche Zielgruppen verfügbar ist. Im speziellen sei dabei auf das Lehrbuch „Adventures in Modeling“ (Colella et al. 2001) hingewiesen. Im Folgenden werden für StarLogo die wichtigsten Elemente für die Simulation sozialer Phänomene vorgestellt. Diese Elemente werden anschließend zu einer Simulation der Diffusion einer Information in einer Population von 100 Agenten zusammengestellt.

StarLogo

StarLogo ist eine “programmierbare Modellierungsumgebung zur Erforschung der Arbeitsweise von dezentralisierten Systemen“ [2]. Das am Massachusetts Institute of Technology (MIT) entwickelte Programm ist eine Weiterentwicklung der Programmiersprache Logo. Mit der ursprünglichen Logo Programmiersprache (eine Windows Version ist z.B. MSW Logo[3] aus dem Jahr 2002) können Nutzerinnen und Nutzer, die keine Programmiererfahrungen haben, durch einfache Befehle ein Symbol auf dem Bildschirm bewegen. Dieses Symbol, das zwar keiner Schildkröte gleicht, aber als „turtle“ bezeichnet wird, hinterlässt eine gezeichnete Spur seiner Bewegungen. So können einfache, aber auch komplexere geometrische Figuren entstehen. Dieses Grundprinzip, dass ein Element mit einfachen Befehlen über den Bildschirm bewegt wird, wird in StarLogo um den Aspekt erweitert, dass es nicht ein Element gibt, sondern viele. StarLogo ist demnach eine Umgebung für Multiagentensimulationen.

Abbildung 2: Turtles aus dem StarLogo Lehrbuch (Colella et al. 2001)

NetLogo[4] kann als Weiterentwicklung von StarLogo betrachtet werden, obwohl StarLogo und NetLogo zwei unterschiedliche Programme von unterschiedlichen Entwicklungsgruppen sind. Die Aufmachung von NetLogo ist attraktiver und die Modellbibliothek ist sehr viel umfangreicher gestaltet. Diese beiden Programme werden in dieser Aufzählung gemeinsam genannt, da sie in der Anwendung fast identisch sind. Der Fokus der Beschreibung liegt in diesem Abschnitt auf dem Programm StarLogo. Ein Argument für den Einsatz von StarLogo als Instrument zum Erlernen von Multiagentensimulationen ist, dass die Entwicklung dieser Software sehr stark vom direkten Einsatz im didaktischen Kontext motiviert ist. So schreiben die Autoren in ihrem Lehrbuch, dass

2 Programm „StarLogo“, Version 2.21, 2008; Quelle: http://education.mit.edu/starlogo/ [10.10.2008].

3 Programm „MSW Logo“, Version 6.5b, 2002; Quelle: http://www.softronix.com/logo.html [10.10.2008].

4 Programm „NetLogo“, Version 4.0.4, Quelle: http://ccl.northwestern.edu/netlogo/ [10.1.2009].

StarLogo entwickelt wurde „um Menschen zu befähigen, ihre eigenen Modelle komplexer, dynamischer Systeme zu bauen“ (Colella et al. 2001).

Beginnen wir den einführenden Ausflug in die Welt von StarLogo mit der Installation der Software auf Ihrem lokalen Computer. Das Auffinden der Software im Internet verläuft ohne Schwierigkeiten. Die Projekthomepage taucht in allen gängigen Suchmaschinen unter den ersten Treffern beim Suchwort „StarLogo“ auf. Der Downloadbereich der Software findet sich auf der Homepage und bietet Downloadversionen für Windows, Macintosh und Unix-Systeme. Der Download erfordert eine Bekanntgabe von Name und E-Mail Adresse und beinhaltet in der Version für Windows einen automatischen Installer, sodass die Installation von StarLogo ohne irgendwelche Vorkenntnisse möglich und mit keinem zusätzlichen Aufwand verbunden ist.

Für den selbständigen Einstieg in StarLogo bieten sich zwei Varianten an. Zum einen über die von StarLogo bereitgestellte Dokumentation, die im Rahmen der Installation lokal eingerichtet wird, aber auch online auf der Projektseite verfügbar ist. Eine andere Variante, die ersten Schritte in StarLogo vorzunehmen, die sich vor allem für den Einsatz im Unterricht oder für das Lernen von StarLogo in Gruppen eignet, ist das von den Autoren des Programms geschriebene Lehrbuch zu StarLogo (Colella et al. 2001).

Praktische Einführung in StarLogo

Im Folgenden wird der Versuch unternommen, eine Einführung in StarLogo unter besonderer Berücksichtigung der für die Simulation sozialer Phänomene notwendigen Elemente zu geben. Dabei soll die Diffusion einer Information ausgehend von wenigen Elementen durch die gesamte Population simuliert werden. Beginnen wir jedoch zuerst mit einem kurzen Überblick über die verwendeten Begriffe. „Turtles“ werden in StarLogo nicht nur Schildkröten genannt. Diese Bezeichnung wurde aus dem ursprünglichen Programm Logo übernommen und steht ganz allgemein für Agenten bzw. Akteure, egal ob Ameisen, Moleküle oder Menschen simuliert werden. Die Anordnung der Simulationen in StarLogo erfolgt auf einem Raster, der die Welt darstellt (siehe Punkt „1“ der folgenden Abbildung). Die einzelnen Felder dieses Hintergrunds werden als „Patches“ bezeichnet. Was die Agenten in unserer Simulation zu tun haben, wird ihnen mittels Befehlen übermittelt. Diese Befehle können entweder einzeln direkt im „Command Center“ eingegeben werden (Punkt „2“ der Abbildung) und als Prozeduren im „Turtle Procedures“ Fenster (Punkt „3“ der Abbildung). Unter Prozeduren versteht man eine Folge zusammenhängender Befehle. Aber dazu später mehr. Befehle müssen weiters in Turtles- und Observer-Befehle unterteilt werden. Observer-Befehle umfassen alle Vorgänge, die nicht im Einflussbereich einzelner Agenten stehen, wie zum Beispiel das Erzeugen von Agenten oder statistische Auswertungen über alle Agenten hinweg. Das Umschalten zwischen Turtles- und Observer-Modus erfolgt mit den Rasterblättern links oben im Kontrollzentrum (Punkt „4“ der Abbildung).

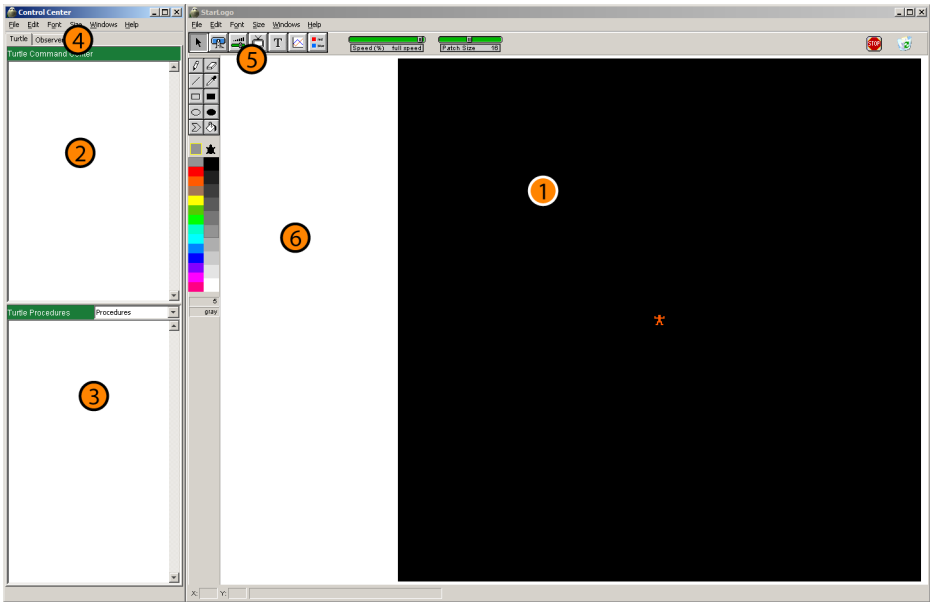

Abbildung 3: Übersicht der Bereiche in StarLogo

Die vorangehende Abbildung zeigt in der rechten Hälfte die Simulationswelt von StarLogo. In der Mitte dieser Welt kann man nach dem Start der Software schon den ersten Akteur erkennen. Tatsächlich handelt es sich dabei nicht um einen Akteur, sondern um 100 Akteure, die auf dem gleichen Feld übereinander liegen. Da im Normalfall das Symbol für die Akteure standardmäßig eine Schildkröte ist und für die Simulation sozialer Phänomene Personen besser geeignet sind, soll die Umstellung der Darstellung der Akteure der erste Befehl sein, den Sie im Rahmen dieser Einführung kennen lernen. Schreiben sie dazu links oben in das „Command Center“ im Turtle-Modus folgenden Befehl und bestätigen Sie diesen Befehl mit der Return Taste:

```
setshape person-shape
```

Die Darstellung des in der Mitte der Simulationswelt dargestellten Akteurs sollte sich jetzt in ein kleines Männchen verändert haben. Mit dem nächsten Befehl bringen wir schon Bewegung in unsere Simulationswelt. Geben Sie die folgende Zeile wieder in das „Command Center“ im Turtles-Modus ein und bestätigen Sie mit Drücken der Return Taste.

```
fd 20
```

Die Abkürzung fd steht für forward und bewirkt, dass sich jeder Akteur 20 Schritte nach vorne bewegt. In welche Richtung sich ein Akteur bewegt, hängt davon ab, in welche Richtung der einzelne Akteur gerade ausgerichtet ist. Um diese Richtung zu ändern gibt es einerseits den Befehl „right turn“, der in der StarLogo Welt wie folgt abgekürzt wird:

```
rt 180
```

Die Zahl hinter rt steht dabei für die Anzahl der Grade, mit denen jeder Akteur sich nach rechts drehen soll. Die oben getätigte Angabe von 180 Grad bewirkt also, dass jeder Akteur sich umdreht und wieder zur Mitte des von den Akteuren erzeugen Kreises ausgerichtet ist. Mit dem Befehl left turn,

```
lt 45
```

drehen sich die Akteure um 45 Grad nach links. Ganz nebenbei haben Sie mit den ersten Befehlen eine zentrale Eigenschaft von StarLogo kennengelernt: Jeder Befehl und jede Prozedur gilt für alle Akteure gleichzeitig.

Im nächsten Schritt werden wir die Verwendung von Prozeduren kennenlernen, damit zusammenhängende Befehle an die Akteure nicht jedes Mal neu eingegeben werden müssen. Geben Sie also links unten in das „Turtles-Procedures“ Fenster (Punkt „3“ in der Abbildung) folgende Zeilen ein:

```
to go
  rt 10
  fd 1
end
```

Die Einrückungen der zweiten und dritten Zeile sind nicht zwingend, erhöhen aber die Lesbarkeit des Programms und werden daher dringend empfohlen. Ob diese Einrückungen mit Tabulator oder Leerzeichen passieren bleibt jedem selbst überlassen. Wenn Sie jetzt links oben im „Turtle Command Center“ den Befehl

```
go
```

eingeben und durch Drücken der Return Taste ausführen, werden alle Befehle ausgeführt, die in der Prozedur go definiert wurde: eine Rechtsdrehung um 10 Grad und ein Schritt nach vorne. Damit das Gefühl einer automatisch ablaufenden Simulation entstehen kann werden wir im nächsten Schritt aus der Symbolleiste (Punkt „5“ der Abbildung oben) den zweiten Button („Create or edit a button“) auswählen und danach in die weiße Fläche neben der Simulationswelt (Punkt „6“ in der Abbildung oben) klicken. Dadurch wird ein neuer Button erzeugt und das Fenster mit den Eigenschaften des Buttons öffnet sich. Geben Sie jetzt in das Feld „StarLogo Instructions“

```
go
```

ein und klicken neben „Forever?“ das Häkchen an. Nach Beenden mit OK ist der Button in der Arbeitsfläche eingefügt und kann angeklickt werden. Aufgrund der vorher definierten Eigenschaften des Buttons (das Häkchen bei „Forever?“), wird die Prozedur solange ausgeführt, bis der Button erneut gedrückt wird. Das Ändern oder Löschen eines Buttons oder eines anderen Elements in diesem Bereich ist nicht sehr intuitiv und passiert indem mit der Maustaste durch Drücken und Ziehen ein Bereich ausgewählt wird, der das zu bearbeitende Element enthält. Danach kann durch Drücken der Entfernen-Taste das markierte Element gelöscht werden oder durch Doppelklick auf das Element bearbeitet werden.

In realen sozialen Systemen agieren die Akteure im Normalfall nicht so synchron wie in unserem Beispiel, sondern wandern scheinbar individuell durch die Welt. Dieser Effekt wird in Multiagentensimulationen in der Regel durch das Hinzufügen von zufälligen Handlungen erreicht. Der Befehl, der eine Zufallszahl erzeugt wird z.B. mit

```
random 5
```

ausgeführt, wobei 5 die nicht erreichte Obergrenze des Bereichs ist, aus dem die Zufallszahl erzeugt wird. Die untere Grenze ist stets 0. Der Befehl von oben erzeugt also zufällig eine Zahl aus der Gruppe 0, 1, 2, 3 oder 4. Ändern wir die vorher erstellte Prozedur in

```
to go
  rt random 30
  lt random 30
  fd 1
end
```

bewegen sich die 100 Akteure unserer ersten kleinen Simulation schon recht authentisch durch die Simulationswelt. Wie Sie an diesem Beispiel sehen, wird der Befehlt random in Verbindung mit einer Zahl dort verwendet, wo eine Zahlenangabe erwartet wird. Wenn Sie diese Simulation einige Sekunden laufen lassen, sehen Sie eine weitere zentrale Eigenschaft von StarLogo Simulationen: Die Welt hat keine Ränder. Akteure, die den dargestellten Bereich oben verlassen, erscheinen am unteren Rand wieder und umgekehrt. Ebenso sind der rechte und der linke Rand miteinander verbunden. Das dabei entstehende geometrische Element wird als Ringtorus bezeichnet.

In den nächsten Schritten werden wir die fehlenden Elemente hinzufügen, um die Diffusion einer Information simulieren zu können. Zuerst betrachten wir die farbliche Erscheinung der Akteure. Diese wird bei der Erzeugung der Akteure automatisch vergeben und führt zu einer sehr bunten Simulationswelt. Um eine zentrale unterschiedliche Eigenschaft, wie z.B. besitzt ein Akteur eine Information oder nicht, darstellen zu können, bietet sich die Projektion dieser Eigenschaft auf die Akteursfarbe an. Dazu müssen jedoch zuerst sämtliche Akteure auf eine gemeinsame Farbe gesetzt werden. Mit

```
setcolor gray
```

im „Turtle Command Center“ eingegeben verändern sämtliche Akteure ihre Farbe auf grau. Im nächsten Schritt ist es notwendig, eine bestimmte Ausgangsanzahl von Akteuren auf eine andere Farbe, z.B. rot, zu setzen, um zu simulieren, dass diese Träger der Information sind. Um zufällig 10 Prozent der Akteure rot zu färben bedienen wir uns neben dem eigentlichen Befehl zum Ändern der Akteursfarbe eines zentralen Befehls im Zusammenhang mit Simulationen, der If-Bedingung:

```
if (10 > random 100) [setcolor red]
```

Die If-Bedingung ist in StarLogo wie oben angeführt definiert. Hinter dem Befehl if folgt eine runde Klammer, welche die Bedingung enthält. In unserem Fall wird eine Zufallszahl kleiner 100 erzeugt, also eine Zahl aus der Menge 0, 1, 2, … 98, 99. Diese Zufallszahl wird dann mit der Zahl 10 verglichen. Ist diese Zahl kleiner als 10, wird der Befehl in der eckigen Klammer dahinter ausgeführt. Trifft die Bedingung in den runden Klammern nicht zu, wird kein Befehl ausgeführt. Die Befehlszeile oben führt also dazu, dass für jeden Akteur mit einer 10%-igen Wahrscheinlichkeit die Farbe auf rot geändert wird. In unserer Startverteilung sind daher ca. 10% der Akteure rot und der Rest grau gefärbt.

Um diese und auch weitere Setup-Einstellungen vorzunehmen ist es üblich, eine Setup-Prozedur zu erstellen und im Fall von StarLogo mit einem Button zu verbinden. Im Fenster für Turtle Prozeduren, in dem sich bereits die go Prozedur befindet fügen wir jetzt eine zweite Prozedur hinzu:

```
to setup_turtle
  rt random 360
  jump random 100
  setcolor gray
  if (10 > random 100) [setcolor red]
end
```

Der einzig neue Befehl in dieser Prozedur ist der jump-Befehl, dieser ist vergleichbar mit dem forward-Befehl fd, allerdings mit dem Unterschied, dass keine 100 Schritte gesetzt werden, sondern die Akteure direkt zum Ziel springen. Zusammengefasst erledigt unsere Setup-Prozedur, dass jeder Akteur zufällig irgendwo in der Simulationswelt positioniert wird, dass alle Akteure grau eingefärbt werden und dann für ca. 10 % die Farbeigenschaft auf rot gesetzt wird. Analog zur Erstellung des Buttons für die go-Prozedur fügen wir jetzt einen Button hinzu und schreiben in das Feld „StarLogo Instructions“

```
setup_turtle
```

Das Häkchen bei „Forever?“ wird diesmal nicht gesetzt, da diese Prozedur nicht wiederholt, sondern nur einmal ausgeführt werden soll. Im nächsten Schritt wird die Interaktion zwischen den Akteuren hinzugefügt. Dazu verwenden wir das Konstrukt

```
color-of one-of-turtles-here
```

das als Wert die Farbe eines anderen Akteurs zurückliefert, der sich auf dem gleichen Feld der Simulationswelt befindet. In Kombination mit der oben vorgestellten If-Bedingung und dem Einfärben des Akteurs können wir folgenden Befehl zusammenstellen:

```
if ((color-of one-of-turtles-here) = red) [setcolor red]
```

Natürlich passiert nach Eingabe dieses Befehls im „Turtle Command Center“ nicht viel, da sich der Großteil der Akteure alleine auf einem Feld der Simulationswelt befinden. Diese If-Bedingung muss natürlich nach jedem Schritt unserer Akteure ausgeführt werden. Fügen Sie deshalb diese Zeile als neuen letzten Befehl in die go Prozedur ein. Nachdem der Akteur zufällig seine Richtung ändert und einen Schritt nach vorne gegangen ist, wird mit dem neuen Befehl ermittelt, ob sich ein anderer Akteur auf dem gleichen Feld aufhält und ob dieser rot eingefärbt ist. Ist dies der Fall wird vom jeweiligen Akteur die Farbe übernommen und ebenfalls auf rot gesetzt. Da wir die Farbe der Akteure als Metapher für die Verbreitung von Informationen verwenden, müssen Akteure also auf die Farbe von anderen Akteuren „schauen“. Um die Verbreitung übersichtlich beobachten zu können werden wir jetzt noch mit dem 6. Symbol von links in der Symbolleiste ein Plot-Fenster hinzufügen. Nach Auswahl des Line Chart Typs erscheint das leere Plot-Fenster. Im Menü findet man unter Format/Data die Einstellungen für Auswertungen. Standardmäßig sollte hier bereits „Number of Turtles“ ausgewählt sein. Durch Klicken auf das Auswahlfeld neben dem Wort „with“ wird rechts daneben ein Eingabefeld aktiviert. Fügen Sie in dieses anstelle des Wortes „true“ die folgende Bedingung ein:

```
color = red
```

Die so erstellte Grafik zeigt uns die Anzahl der roten Akteure einmal pro Sekunde an. Im Plot-Menü kann zu einem späteren Zeitpunkt unter Edit/Clear Plot die bisherige Ausgabe gelöscht werden. Bestätigen Sie das offene Fenster mit „OK“ und fertig ist die Simulation. Starten Sie die Simulation durch einmaliges Drücken des setup_turtle-Button und des go-Button.

Um unsere Simulation durch Interaktion ein wenig komfortabler und sie auch für andere Menschen zugänglich zu gestalten fügen wir noch das eine oder andere interaktive Element hinzu. Die folgenden Schritte bewirken keine Veränderungen der bisher erstellten Simulation. Sie können hier die Übung auch beenden und im nächsten Abschnitt weiterlesen. Ich empfehle Ihnen aber, sich auch die weiteren Interface-Elemente von StarLogo anzueignen, da diese mit wenigen Handgriffen Ihre Simulationen auf ein professionelles Niveau heben. Als Erstes fügen wir eine Setup-Funktion hinzu, die alle Akteure löscht und neu platziert. Zu diesem Zweck wechseln wir zum ersten Mal in den Observer-Modus durch Drücken des Rasterblattes „Observer" links oben im Kontrollzentrum (Punkt „4" der Abbildung) und fügen eine Observer Prozedur hinzu:

```
to setup_observer
  ca
  crt 100
end
```

Der erste Befehl ca steht für clear all und löscht die gesamte Simulationswelt. Der zweite Befehl steht für create turtles und erzeugt in dieser Form 100 neue Akteure, die im Zentrum der Simulationswelt angeordnet werden. Fügen Sie jetzt wie bereits oben zweimal durchgeführt einen Button ein und geben Sie in das Feld „StarLogo Instructions"

```
setup_observer
```

ein. Dieses Mal müssen Sie rechts das Optionsfeld „Observer" anklicken um StarLogo mitzuteilen, dass dieser Button eine Observer-Prozedur aufruft. Weitere Elemente der Interaktivität stellen Slider dar. Fügen Sie ein Slider Element (das dritte Symbol von links) aus der Symbolleiste in die Steuerungsfläche (Punkt „6") hinzu und tragen Sie im Feld „Variable"

```
anzahl_akteure
```

ein. Bei „Minimum" fügen Sie bitte 10 ein, bei „Maximum" 200 und bei „Current" 100. Dieser Slider dient der Einstellung, wie viele Akteure für eine Simulation verwendet werden sollen, die Werte für Ober- und Untergrenze können später beliebig geändert werden. Damit die Anzahl der zu erzeugenden Akteure bei der Betätigung des setup_observer Buttons tatsächlich von der aktuellen Einstellung des Sliders abhängt, ändern wir die Zeile in der Observer Prozedur, in der die Akteure erzeugt werden wie folgt ab:

```
crt anzahl_akteure
```

Jetzt wird die Variable aus dem Slider ausgelesen und diese Zahl dem Befehl zur Erzeugung von Akteuren übergeben. Ein zweiter Slider bietet sich für den Anteil der in der Population ursprünglich mit der Information ausgestatteten Akteure an. Fügen wir also einen weiteren Slider hinzu, schreiben in das Feld „Variable":

```
start_prozent
```

und geben als „Minimum" den Wert 0, als „Maximum" den Wert 99 und als „Current" den Wert 10 an. Um ähnlich wie bei dem vorangegangenen Slider auf die Variable des Sliders zugreifen zu können ändern wird in der setup_turtle Prozedure die Zeile mit dem If-Befehl wie folgt ab:

```
if (start_prozent > random 100) [setcolor red]
```

Die folgende Abbildung zeigt Ihnen abschließend das Ergebnis des einführenden Beispieles (schön ist die S-Kurve der Diffusion zu sehen) und im Anschluss findet sich zusammengefasst der Code[5] der für diese Simulation verwendeten Prozeduren.

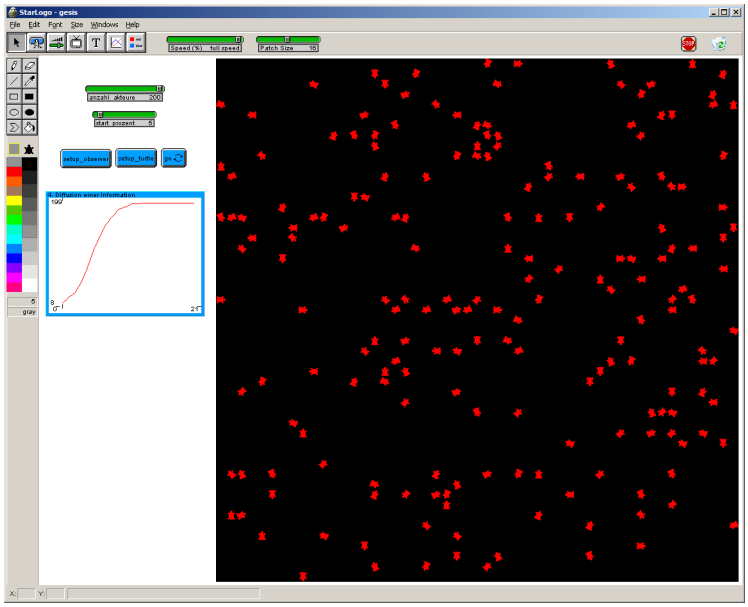

Abbildung 4: Die fertige Simulation

Turtle Prozeduren:

```
to go
  rt random 30
  lt random 30
  fd 1
  if ((color-of one-of-turtles-here) = red) [setcolor red]
end

to setup_turtle
  rt random 360
  jump random 100
  setcolor gray
  if (start_prozent > random 100) [setcolor red]
end
```

5 Eine StarLogo-Datei mit der gesamten Simulation finden Sie unter http://www.pfeffer.at/soFid [30.1.2009]

Observer Prozedur:

```
to setup_observer
  ca
  crt anzahl_akteure
end
```

Natürlich werden für die Modellbildung realer sozialer Systeme komplexere Vorgänge abzubilden sein. StarLogo kann viele dieser Vorgänge mit einfachen Befehlen in eine Simulation umsetzen. Neben den Turtles, also den Akteuren der Simulationswelt, können z.B. auch die Patches (der Hintergrund) programmiert werden und als Interaktion für die Akteure zur Verfügung stehen. So ist es zum Beispiel möglich, dass Hindernisse oder bestimmte Zonen definiert werden, die auf darauf befindliche Akteure eine Auswirkung haben. Beispielsweise „finden" Akteure Elemente und transportieren diese an einen anderen Ort, usw. Weiters ist es möglich, dass Akteure mehrere Eigenschaften mit sich tragen, z.B. einen Wert, der angibt, mit welcher Wahrscheinlichkeit die Information weitergegeben wird, oder Werte für demographische Eigenschaften. StarLogo beinhaltet standardmäßig einige Beispielsimulationen aus unterschiedlichen Bereichen. Da der Code der Beispielmodelle offen ist, können diese zum Experimentieren und damit zum Lernen des Umgangs mit StarLogo sehr gut verwendet werden. In NetLogo findet sich zudem noch eine weitaus größere Ansammlung von Beispielsimulationen aus dem Bereich sozialer Phänomene: Modelle zur Simulation der Verbreitung von Gerüchten, das Verhalten von Party-Gästen, aber auch die Modellierung von Phänomenen in sozialen Netzwerken, usw.

Übersicht Simulationsumgebungen

Im Folgenden findet sich eine Übersicht über verschiedene Eigenschaften von drei Simulationsumgebungen für Multiagentensimulationen, welche den Einstieg in diese Art der Simulation vereinfachen soll. StarLogo bzw. NetLogo sind hervorragende Instrumente für den Einstieg in eigene Projekte. AnyLogic ist die Profiversion, die, was die Anzahl der Akteure betrifft nach oben fast unbegrenzte Möglichkeiten bietet. Der Nachteil von AnyLogic ist der hohe Preis. Repast ist der Versuch die Vorteile von AnyLogic in einer Open-Source Software anzubieten. Allerdings erfordert Repast fortgeschrittene Programmierkenntnisse.

Tabelle 1: Übersicht von Simulationsumgebungen für Multiagentensimulationen

	StarLogo/NetLogo	Repast	AnyLogic
Quelle	http://education.mit.edu/starlogo http://ccl.northwestern.edu/netlogo/	http://repast.sourceforge.net	http://www.xjtek.com/ anylogic
Zugang, Kosten	kostenloser Download von der Projekthomepage	kostenloser Download bei Sourceforge, open-source	kostenpflichtig (Educational Version $ 330,-), 15 Tage gratis Testversion
Erste Schritte	gute Einführung, sehr einfach funktionierende Beispiele	Beispielprojekt für Anfänger Innen, dennoch sehr aufwändig	gut nachvollziehbar, trotz kompliziertem Programm
Dokumentation	online und lokal installiert vorhanden	vorhanden, auch FAQ Sammlung vorhanden	online und als Teil des Programms
Modellbibliothek	Beispiele aus verschiedenen Bereichen werden lokal installiert, leicht zu starten	Beispielprojekte vorhanden aber schwer zu finden, Installation erfordert Eclipse Kenntnisse	ca. 50 Modellbeispiele in Programm über Startmenü, gute Beschreibung, leicht zu starten
Motivation, Erfolgserlebnisse	schnelle Erfolgserlebnisse, motiviert zum ausprobieren	für UserInnen ohne Erfahrungen mit Eclipse-Umgebung demotivierend	viele Schritte vor eigentlicher Simulation notwendig, dennoch erfolgreiche Umsetzung möglich
Gesamteindruck	einfach gehaltenes Tool, nicht für sehr große oder komplexe Simulationen geeignet	sehr aufwändiges professionelles Tool, das nicht für den schnellen Einstieg geeignet ist	anspruchsvolles und professionelles Tool, eigene umfassende Simulationswelt

Conclusio

Modellbildungsprozesse sind gleichzeitig Konstruktions- und Abbildungsprozesse. Die ModellbauerInnen trennen durch Abstraktion das Wesentliche vom Unwesentlichen und konstruieren damit aus dem Blickwinkel des Konstruktivismus einen Entwurf eines realen Systems. Zudem sind Modelle im erkenntnistheoretischen Sinn Abbilder der Wirklichkeit, da diese Eigenschaften des abzubildenden Objekts beinhalten. Die Menschen spiegeln ihre Umwelt zunächst geistig wider, indem sie Bilder und Zusammenhänge des Wahrgenommenen, mentale Modelle, im Kopf erzeugen. Dabei handelt es sich um vereinfachte, weniger komplexe und oft auch verfälschte Abbildungen des realen Systems. Diese Widerspiegelungsprodukte sind nie eine objektive Wiedergabe der „Realität“ (zu der es keinen direkten Zugang gibt), sondern immer gleichzeitig *Abbildung* und *Entwurf*, also menschliche Konstruktionen bestimmter Aspekte der Umwelt. In diesen Konstruktionen finden die Rahmenbedingungen der Menschen ihren Niederschlag, es gehen die bisherigen Erfahrungen der Einzelnen genauso ein wie deren Interessenslagen und Lebensbedingungen. Durch Interaktion mit anderen Menschen oder mit der sonstigen Umwelt kann sich die Sicht der Dinge durchaus verändern. Die Konstruktionen sind daher im Zeitverlauf nicht unbedingt invariant, sondern die Sicht der Dinge ist potentiell variabel.

Die Welt zu beschreiben, kann mitunter ein sehr komplexes Unterfangen sein. Wenn das zu beschreibende System noch dazu ein soziales ist, ist in vielen Fällen eine zufriedenstellende umfassende Beschreibung nicht möglich, da unzählige Dimensionen ineinander greifen und einander beeinflussen. Diese Beschreibung des realen Systems ist jedoch Voraussetzung für die Übertragung in ein computerunterstütztes Modell. Die zentrale Herausforderung bei dieser Übertragung ist es, dass genau so viele Eigenschaften des realen Systems übertragen werden, wie unbedingt zur Beschreibung notwendig sind – und nicht mehr. Keep the model simple.

Die Modellkonstruktion auf der Grundlage von realen Systemen und die darauffolgenden Simulationen sind nicht an naturwissenschaftliche Phänomene gebunden. In jedem Fachgebiet können Aspekte in Computersimulationen nachgebildet werden, so auch im Bereich der Sozialwissenschaften. Simulationsumgebungen, wie z.B. StarLogo, ermöglichen auch Forscherinnen und Forschern, die keine Ausbildung zum Erstellen von Computerprogrammen haben, Simulationen zu ihren Forschungsfragen zu erzeugen. Das in diesem Artikel vorgestellte Einführungsbeispiel zeigt, dass mit StarLogo Multiagentensimulationen mit wenig Aufwand und mit wenigen Befehlen erstellt werden können.

Literaturverzeichnis

Colella, Vanessa S., Eric Klopfer und Mitchel Resnick, 2001: Adventures in Modeling, Exploring Complex, Dynamic Systems with StarLogo. New York: Teachers College Press.

Gilbert, Nigel und Klaus G. Troitzsch, 2005: Simulation for the Social Scientist, 2nd. Edition. New York: Open University Press.

Gilbert, Nigel, 2007: Agent-Based Models, Series: Quantitative Applications in the Social Sciences. Thousand Oaks: Sage Pubn Inc.

Gordon, Deborah, 1999: Ants at Work, How an Insect Society is Organiszed. New York: W. W. Norton & Company Ldt.

Johnson, Steven, 2001: Emergence. London: The Penguin Press.

Kirkup, Les, 1994: Experimental Methods – An Introduction to the Analysis and Representation of Data. Brisbane: John Wiley & Sons.

Kühl, Stefan, 2005: Experiment. In: Stefan Kühl, Petra Strodtholz und Andreas Taffertshofer (Hrsg.), Quantitative Methoden der Organisationsforschung. S. 213-242. Wiesbaden: VS Verlag.

Miller, John H. und Scott E. Page, 2007: Complex Adaptive System – An Introduction to Computational Models of Social Life. Princeton: Princeton University Press.

Resnick, Mitchel, 1994: Turtles, Termites, and Traffic Jams. Cambridge: The M.I.T. Press.

Stachowiak, Herbert, 1973: Allgemeine Modelltheorie. Wien: Springer-Verlag.

VDI-Gesellschaft Fördertechnik Materialfluss Logistik (Hrsg.), 1996: VDI-Richtlinie: VDI 3633, Simulation von Logistik-, Materialfluß- und Produktionssystemen – Begriffsdefinitionen.

Zum Autor

Jürgen Pfeffer, Mag. rer. soc. oec, geb. 3. April 1976. Wiener Netzwerkanalytiker; Doktorand an der Technischen Universität Wien, Unternehmensberater mit Schwerpunkt auf netzwerkanalytische Verfahren; Wissenschaftlicher Fokus: Struktur und Dynamik von Mensch-zu-Mensch Kommunikationsnetzwerken.
Details und Kontakt: www.pfeffer.at